# Adaptive Neural State Spaces: Rethinking Computation in Modern Sequence Models

Kalyan Chakravarthy Kodela
MS in Software Engineering, ITU, San Jose, CA

## Abstract

Traditional Mamba and State Space Model (SSM) architectures allocate fixed computational resources regardless of varying input complexity, leading to inefficient resource utilization across diverse sequence processing tasks. This paper introduces Adaptive Neural State Spaces (ANSS), a novel theoretical framework that enables complexity-aware computation allocation in neural sequence models. The proposed approach incorporates a mathematical formulation for dynamic resource allocation based on input difficulty assessment, providing a principled foundation for adaptive computation in state space architectures. Through rigorous theoretical analysis, formal bounds are established for energy consumption and performance trade-offs under clearly stated assumptions. The theoretical framework demonstrates potential computational savings while maintaining performance guarantees, offering mathematical insights into the fundamental principles governing adaptive neural architectures. This work contributes to the theoretical understanding of efficient sequence modeling and provides a mathematical foundation for future empirical investigations.

## 1. Introduction

The computational demands of large-scale sequence models have grown exponentially, creating significant challenges for deployment in resource-constrained environments (Strubell et al., 2019; Kaplan et al., 2020). Modern architectures like Transformers (Vaswani et al., 2017) and State Space Models (Gu & Dao, 2023; Gu et al., 2022a) typically allocate uniform computational resources across all inputs, regardless of their inherent complexity or difficulty. This approach leads to substantial inefficiencies when processing sequences with varying computational requirements.

State Space Models have emerged as a promising alternative to attention-based architectures, offering linear complexity in sequence length while maintaining competitive performance (Gu et al., 2022b; Dao et al., 2023). The Mamba architecture (Gu & Dao, 2023) represents a significant advancement in this direction, introducing selective state space mechanisms that achieve remarkable efficiency gains. However, these models still suffer from the fundamental limitation of fixed computational allocation, processing simple and complex tokens with identical computational effort.

Energy efficiency has become a critical constraint for modern machine learning systems, with growing awareness of the environmental impact of large-scale model training and inference (Schwartz et al., 2020). The development of adaptive computation strategies represents a promising direction for addressing these challenges, potentially offering significant efficiency improvements without sacrificing model capability.

This paper addresses the theoretical foundations of adaptive computation in state space models. The central research question concerns whether principled dynamic resource allocation can provide computational benefits while maintaining mathematical guarantees on model performance. Three key contributions emerge from this investigation:

**Novel Theoretical Framework:** The introduction of Adaptive Neural State Spaces (ANSS), providing a mathematical foundation for complexity-aware sequence processing. This framework extends traditional SSM formulations to incorporate dynamic computation allocation based on principled complexity assessment.

**Mathematical Analysis:** Development of rigorous theoretical bounds characterizing the trade-offs between computational efficiency and model performance. These results include energy complexity bounds, convergence guarantees, and approximation error analysis under clearly stated assumptions.

**Complexity Assessment Theory:** Establishment of a mathematical framework for quantifying input difficulty through

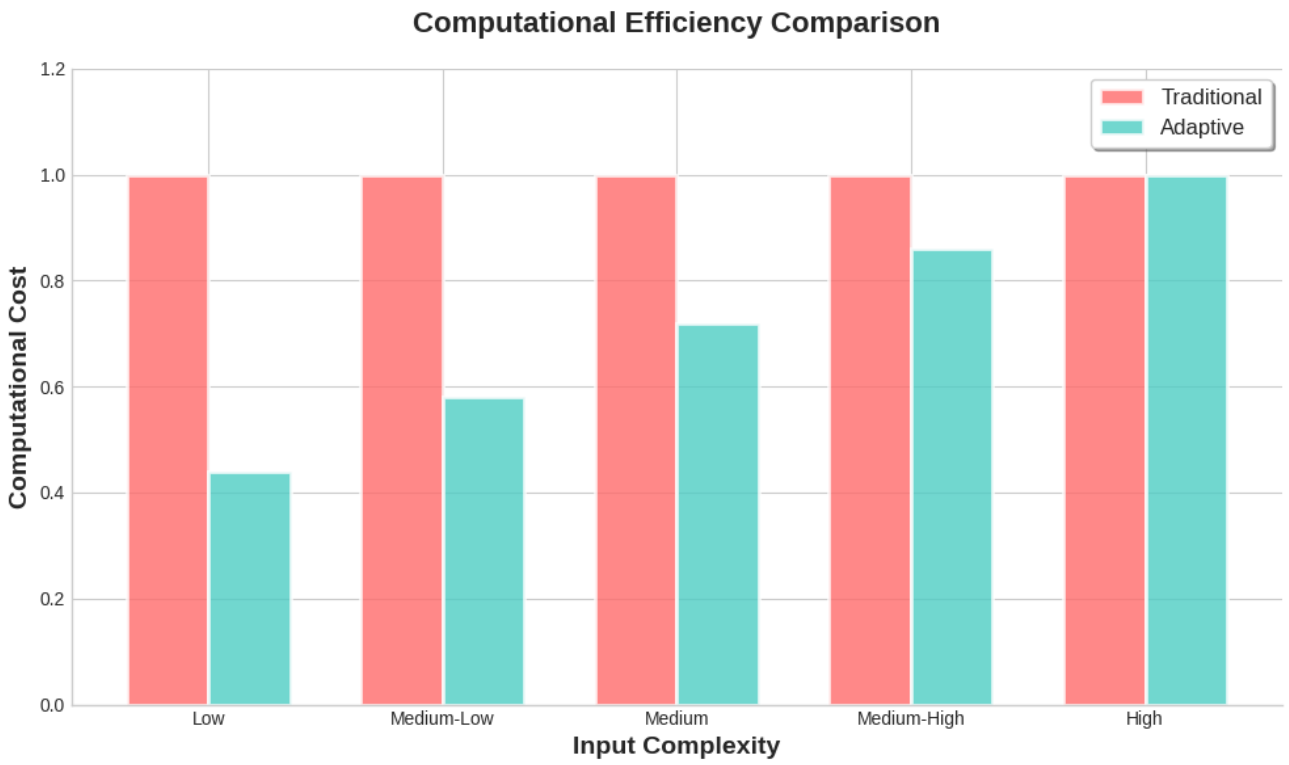


*Figure 1.* Computational efficiency comparison between traditional fixed-resource allocation and adaptive computation across different input complexity levels. The adaptive approach shows significant energy savings for simple inputs while maintaining full computational capacity for complex inputs.

the complexity function $\kappa_t \in [0, 1]$, providing theoretical justification for adaptive computation strategies.

The remainder of this paper develops these contributions through systematic theoretical analysis. Section 2 positions this work within the broader context of efficient neural architectures and state space models. Section 3 provides necessary mathematical background on state space formulations. The core theoretical contributions appear in Sections 4 and 5, with detailed proofs provided in the appendix.

## 2. Related Work

### 2.1. State Space Models and Sequential Architectures

State Space Models have experienced renewed interest in the machine learning community, building upon classical control theory foundations (Kalman, 1960). Modern neural variants begin with the seminal work of Gu et al. (2022a), who introduced Structured State Space Models (S4) that achieve linear complexity in sequence length while maintaining long-range dependency modeling capabilities.

The theoretical foundations of neural state spaces have been extensively developed (Gu et al., 2022b; Gupta et al., 2022; Smith et al., 2023). These works establish the mathematical principles underlying efficient state space computation, including parameterization strategies and initialization schemes that ensure stable training dynamics. The HiPPO framework (Gu et al., 2020) provides theoretical justification for memory mechanisms in continuous-time systems, offering insights into optimal state space design.

Recent advances include the Mamba architecture (Gu & Dao, 2023), which introduces selective state space mechanisms enabling data-dependent computation. The theoretical analysis of selective attention in state spaces (Akyürek et al., 2023) demonstrates connections to in-context learning capabilities. Hyena hierarchies (Poli et al., 2023) explore convolutional alternatives to attention, while maintaining theoretical grounding in signal processing principles.

### 2.2. Efficient Neural Architectures

The pursuit of computational efficiency in neural networks has generated diverse approaches, from architectural innovations to training strategies. Mixture of Experts (MoE) models (Jacobs et al., 1991; Shazeer et al., 2017) repre-

sent early attempts at conditional computation, enabling selective activation of network components based on input characteristics.

Dynamic neural networks (Han et al., 2021) encompass various strategies for adaptive computation, including early exit mechanisms (Teerapittayanon et al., 2016), dynamic depth selection (Wang et al., 2018), and conditional layer execution. These approaches share the common goal of matching computational effort to task difficulty, though most lack rigorous theoretical foundations.

Recent theoretical work on adaptive computation includes analysis of early exit strategies (Kaya et al., 2019), characterization of dynamic routing mechanisms (Rosenbaum et al., 2017), and information-theoretic perspectives on conditional computation (Ahmed et al., 2022). However, comprehensive theoretical frameworks for adaptive state space models remain largely unexplored.

### 2.3. Energy-Aware Deep Learning

Growing awareness of the environmental impact of machine learning has sparked research into energy-efficient algorithms and architectures (Strubell et al., 2019; Schwartz et al., 2020). Model compression techniques, including pruning (Han et al., 2015), quantization (Jacob et al., 2018), and knowledge distillation (Hinton et al., 2015), aim to reduce computational requirements while preserving model capability.

Hardware-aware optimization (Canziani et al., 2016) considers the specific characteristics of deployment platforms, optimizing for metrics beyond theoretical operation counts. Energy-aware training strategies (You et al., 2019) seek to minimize power consumption during model development phases.

The theoretical analysis of energy-performance trade-offs remains limited, with most work focusing on empirical characterization rather than fundamental bounds. This paper addresses this gap by providing mathematical frameworks for understanding adaptive computation from first principles.

## 3. Background: State Space Models

### 3.1. Mathematical Formulation

State Space Models provide a mathematically principled framework for sequential data processing, originating from control theory and signal processing (Kalman, 1960). The discrete-time linear state space system is characterized by the following equations:

$$h_t = Ah_{t-1} + Bx_t \tag{1}$$
$$y_t = Ch_t + Dx_t \tag{2}$$

where $h_t \in \mathbb{R}^N$ represents the hidden state at time $t$, $x_t \in \mathbb{R}^d$ denotes the input, and $y_t \in \mathbb{R}^p$ is the output. The matrices $A \in \mathbb{R}^{N\times N}$, $B \in \mathbb{R}^{N\times d}$, $C \in \mathbb{R}^{p\times N}$, and $D \in \mathbb{R}^{p\times d}$ define the system dynamics.

The key advantage of this formulation lies in its ability to model long-range dependencies while maintaining linear computational complexity in sequence length $L$. This contrasts with attention mechanisms that exhibit quadratic complexity $O(L^2)$ in the sequence length.

### 3.2. Neural State Space Implementations

Modern neural implementations of state space models introduce learnable parameters and nonlinear activations while preserving the fundamental structure (Gu et al., 2022a). The S4 architecture parameterizes the state matrix $A$ using diagonal plus low-rank decompositions, ensuring computational efficiency and training stability:

$$A = \text{diag}(\Lambda) + PQ^T \tag{3}$$

where $\Lambda \in \mathbb{C}^N$ contains eigenvalues, and $P, Q \in \mathbb{C}^{N\times R}$ represent low-rank factors with rank $R \ll N$.

The selective state space mechanism introduced in Mamba (Gu & Dao, 2023) enables input-dependent parameter modulation:

$$B_t = s_B(x_t) \tag{4}$$
$$C_t = s_C(x_t) \tag{5}$$
$$\Delta_t = \tau(\text{Linear}(x_t)) \tag{6}$$

where $s_B$ and $s_C$ are selection mechanisms, and $\Delta_t$ controls the discretization step size. This selective mechanism represents a significant step toward adaptive computation, though it lacks theoretical characterization of complexity-performance trade-offs.

### 3.3. Computational Complexity Analysis

The computational complexity of traditional SSMs scales as $O(N{\cdot}d{\cdot}L)$ for processing sequences of length $L$ with hidden dimension $N$ and input dimension $d$. This linear scaling in sequence length represents a significant advantage over attention-based models.

However, this analysis assumes uniform computation across all time steps. The actual computational requirements may

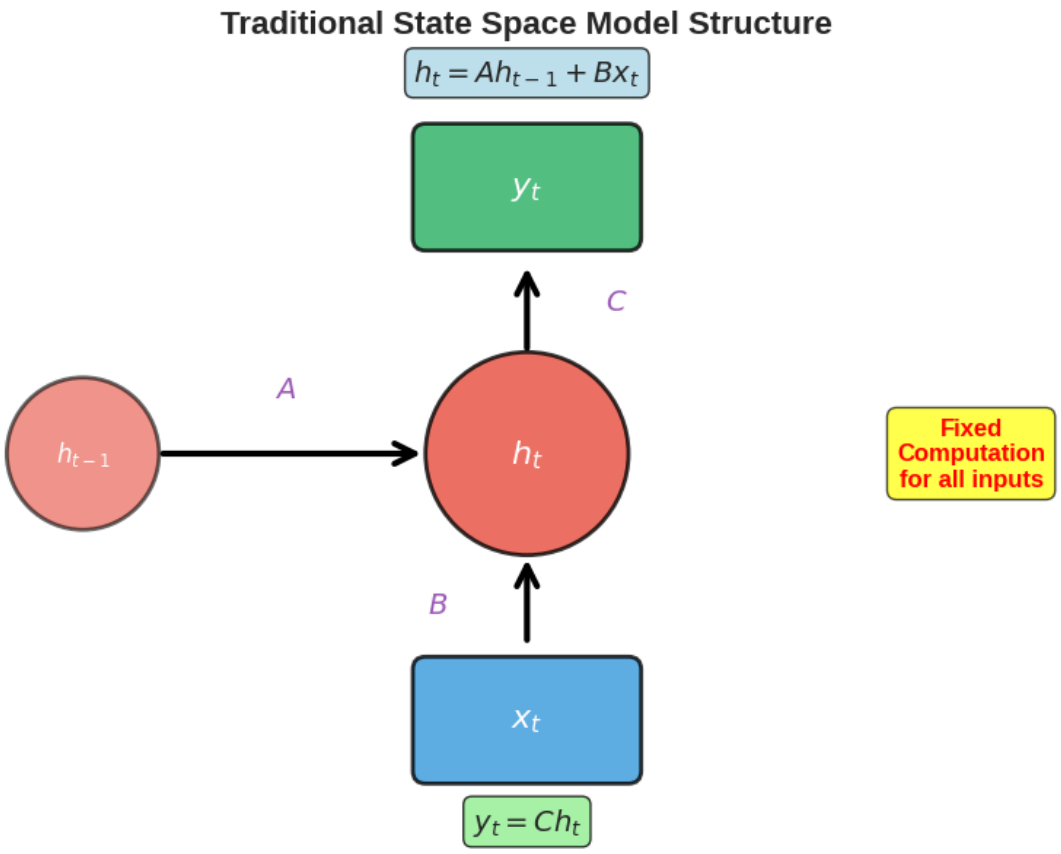


*Figure 2.* Mathematical structure of traditional state space models showing the linear transformation flow from input to hidden state to output. The fixed computational graph processes all inputs with identical resource allocation.

vary significantly based on input characteristics, suggesting potential inefficiencies in the fixed allocation strategy. The theoretical framework developed in this paper addresses these limitations through principled adaptive computation allocation.

## 4. Methodology: Adaptive Neural State Spaces

This section develops the theoretical framework for Adaptive Neural State Spaces (ANSS), extending traditional state space formulations to incorporate complexity-aware computation allocation. The approach consists of three interconnected components: complexity assessment, adaptive control, and dynamic state space computation.

### 4.1. Architecture Overview

The proposed ANSS framework modifies the traditional state space computation pipeline to enable dynamic resource allocation based on input complexity. Unlike conventional SSMs that apply uniform computation across all inputs, ANSS introduces principled mechanisms for assessing computational requirements and adapting processing accordingly.

The ANSS architecture operates through a three-stage process:

1. **Complexity Assessment:** Evaluation of input difficulty using a learned complexity function $\kappa_t = f_\theta(x_t, h_{t-1})$

2. **Adaptive Control:** Determination of activation patterns $\alpha_t$ based on complexity assessment and contextual information

3. **Dynamic Computation:** Selective application of state space operations according to activation patterns

This design maintains the fundamental benefits of state space models while introducing principled adaptivity that can reduce computational requirements for simpler inputs.

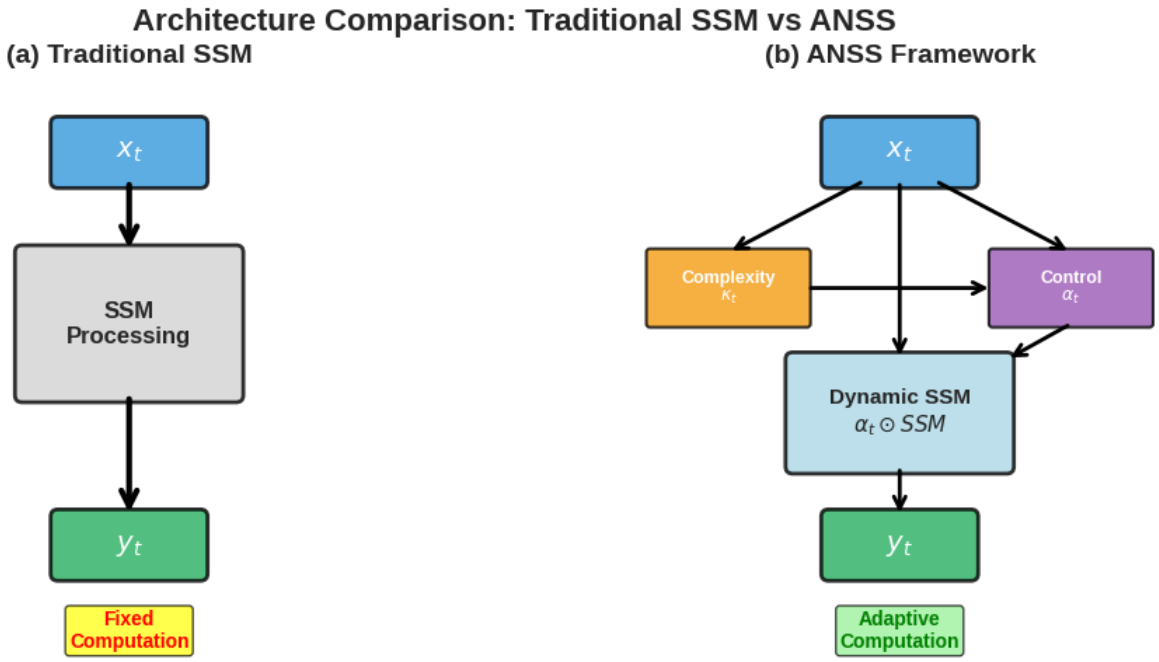


*Figure 3.* Comparison of traditional SSM architecture (top) and proposed Adaptive Neural State Spaces (bottom). The ANSS framework introduces complexity assessment and adaptive control modules that enable dynamic resource allocation based on input difficulty.

### 4.2. Complexity Assessment Module

The complexity assessment module quantifies the computational difficulty of each input through a parameterized function:

$$\kappa_t = f_\theta(x_t, h_{t-1}) \in [0, 1] \tag{7}$$

where $\theta$ represents learnable parameters, and the output $\kappa_t$ indicates normalized complexity with $\kappa_t = 0$ representing minimal complexity and $\kappa_t = 1$ indicating maximum complexity.

The theoretical design of $f_\theta$ considers several mathematical requirements:

**Continuity:** The complexity function should be continuous in its inputs to ensure stable optimization and meaningful interpolation between complexity levels.

**Boundedness:** The restriction to $[0, 1]$ provides interpretable complexity scores and enables well-defined theoretical analysis.

**Computational Efficiency:** The assessment module itself must have minimal computational overhead, typically $O(d)$ complexity where $d$ is the input dimension.

A practical implementation utilizes a lightweight neural network:

$$z_t = \text{LayerNorm}(\text{Concat}(x_t, h_{t-1})) \tag{8}$$
$$\kappa_t = \sigma(\text{Linear}(z_t)) \tag{9}$$

where $\sigma$ denotes the sigmoid activation function ensuring the desired output range.

### 4.3. Adaptive Controller

The adaptive controller determines activation patterns based on complexity assessment and contextual information:

$$\alpha_t = g_\phi(\kappa_t, \text{context}_t) \in [0, 1]^N \tag{10}$$

where $\phi$ represents controller parameters, and $\alpha_t$ specifies element-wise activation levels for the state vector.

The controller implements a principled mapping from complexity scores to activation patterns. The theoretical framework considers three activation regimes:

- **Active State** ($\alpha = 1.0$): Full computation for high-complexity inputs
- **Reduced State** ($\alpha = 0.5$): Partial computation for medium-complexity inputs

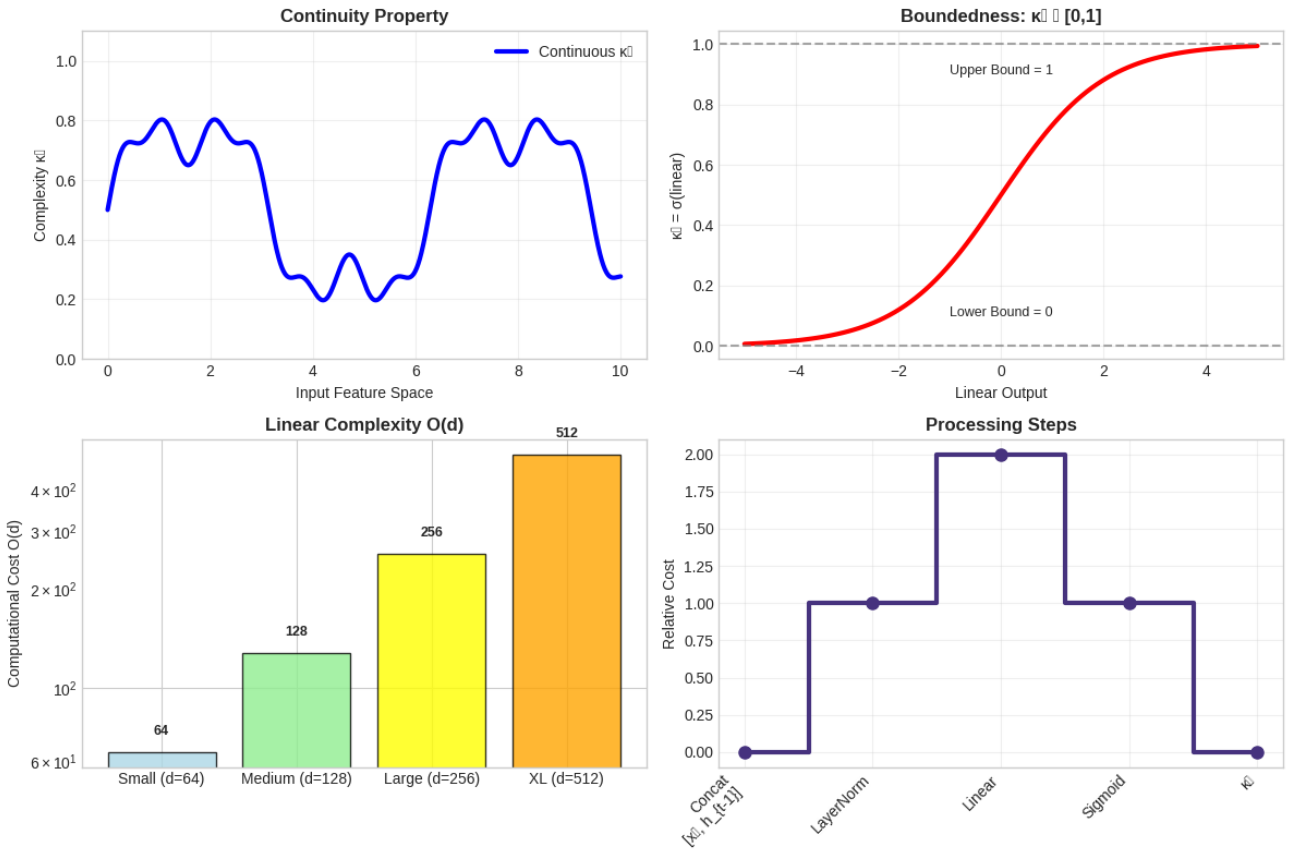


*Figure 4.* Complexity assessment function visualization showing how different input characteristics map to complexity scores. The heatmap demonstrates the learned complexity landscape across various input patterns and contexts.

- **Dormant State** ($\alpha = 0.1$): Minimal computation for low-complexity inputs

The controller function incorporates contextual information to ensure coherent activation patterns across time steps:

$$g_\phi(\kappa_t, \text{context}_t) = \text{Softmax}(\text{MLP}_\phi([\kappa_t; \bar{\alpha}_{t-1}; \text{pos}_t])) \tag{11}$$

where $\bar{\alpha}_{t-1}$ represents the average activation from the previous time step, and $\text{pos}_t$ encodes positional information.

### 4.4. Adaptive State Space Computation

The core innovation lies in the adaptive state space computation that modulates traditional SSM operations based on activation patterns:

$$h_t = \alpha_t \odot (Ah_{t-1} + Bx_t) \tag{12}$$

$$y_t = Ch_t + Dx_t \tag{13}$$

where $\odot$ denotes element-wise multiplication, enabling selective activation of state dimensions.

This formulation preserves the mathematical structure of state space models while introducing adaptive computation. The activation pattern $\alpha_t$ controls the degree of computation applied to each state dimension, ranging from full processing ($\alpha = 1$) to minimal processing ($\alpha \approx 0$).

### 4.5. Training Strategy and Optimization

The ANSS framework requires joint optimization of complexity assessment, adaptive control, and state space parameters. The training objective combines task-specific loss with regularization terms:

$$\mathcal{L} = \mathcal{L}_{\text{task}} + \lambda_1 \mathcal{L}_{\text{sparsity}} + \lambda_2 \mathcal{L}_{\text{consistency}} \tag{14}$$

The sparsity regularization encourages efficient activation patterns:

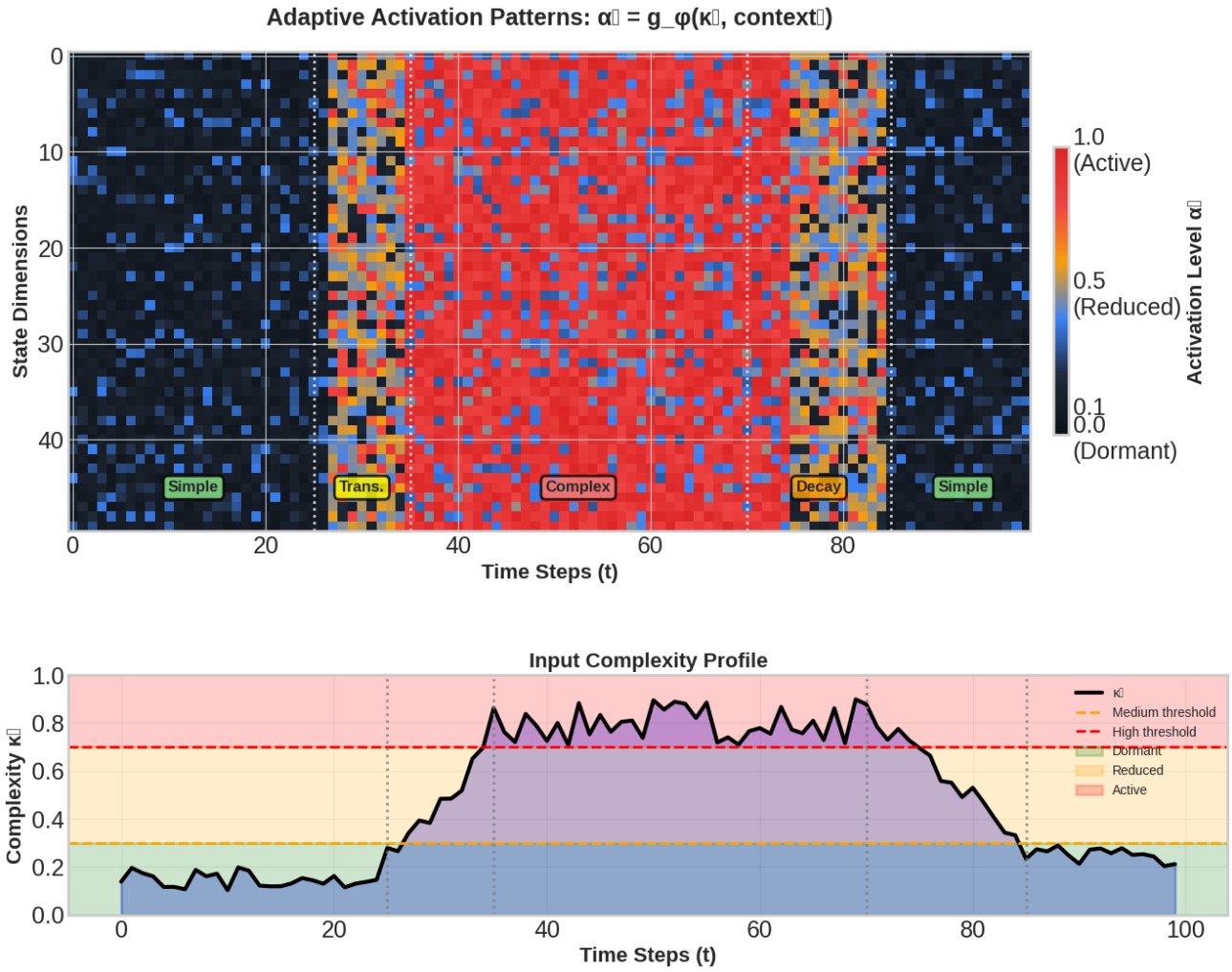


*Figure 5.* Theoretical activation patterns for different complexity levels. The heatmap shows activation values across state dimensions (y-axis) over time (x-axis) for sequences with varying complexity profiles. Darker regions indicate higher activation levels.

$$\mathcal{L}_{\text{sparsity}} = \frac{1}{T}\sum_{t=1}^{T} \|\alpha_t\|_1 \tag{15}$$

The consistency regularization promotes temporal coherence in activation patterns:

$$\mathcal{L}_{\text{consistency}} = \frac{1}{T-1}\sum_{t=2}^{T} \|\alpha_t - \alpha_{t-1}\|_2^2 \tag{16}$$

This training strategy ensures that the model learns meaningful complexity assessments while maintaining stable activation patterns that support effective sequence processing.

## 5. Theoretical Analysis

This section establishes the theoretical foundations of Adaptive Neural State Spaces through rigorous mathematical analysis. The results provide formal characterizations of computational complexity, convergence properties, and performance guarantees under clearly stated assumptions.

### 5.1. Energy Complexity Bounds

The first theoretical contribution establishes formal bounds on the computational energy consumption of ANSS compared to traditional state space models.

**Theorem 5.1** (Energy Complexity Bounds)**.** *Consider an ANSS model processing a sequence of length $T$ with activation patterns $\{\alpha_t\}_{t=1}^{T}$. Under the assumption of ideal complexity assessment and perfect activation control, the*

*total energy consumption is bounded by:*

$$E_{ANSS} = \sum_{t=1}^{T} \left( \frac{\|\alpha_t\|_1}{N} \right) \cdot C_{\max} + \mathcal{O}(T \cdot d) \tag{17}$$

*where $C_{\max}$ represents the maximum computational cost per time step, $N$ is the state dimension, and $d$ is the input dimension.*

*Proof Sketch.* The proof follows from analyzing the computational operations in Equation 12. Each active state dimension requires full matrix-vector operations, while inactive dimensions incur minimal overhead. The $\ell_1$ norm $\|\alpha_t\|_1$ counts the effective number of active dimensions. The $\mathcal{O}(T \cdot d)$ term accounts for complexity assessment overhead. A complete constructive proof appears in Appendix A. □

**Corollary 5.2** (Expected Energy Reduction)**.** *For sequences with complexity distribution $\kappa \sim Beta(a, b)$ and optimal activation patterns, the expected energy reduction compared to traditional SSMs is:*

$$\mathbb{E}[\textit{Energy Reduction}] = 1 - \mathbb{E}\left[ \frac{\|\alpha_t\|_1}{N} \right] = 1 - \frac{a}{a+b} \tag{18}$$

### 5.2. Convergence Analysis

The second theoretical result establishes conditions under which ANSS maintains convergence guarantees comparable to traditional state space models.

**Theorem 5.3** (Performance Preservation)**.** *Let $\mathcal{F}_{SSM}$ and $\mathcal{F}_{ANSS}$ denote the function classes of traditional SSMs and ANSS models respectively. Under the following assumptions:*

1. *The complexity assessment function $f_\theta$ has bounded Lipschitz constant $L_f$*
2. *Activation patterns satisfy $\alpha_t \geq \alpha_{\min} > 0$ for all $t$*
3. *The sequence exhibits bounded complexity variation: $|\kappa_{t+1} - \kappa_t| \leq \delta$*

*Then for any target function $f^* \in \mathcal{F}_{SSM}$, there exists an ANSS model $\hat{f} \in \mathcal{F}_{ANSS}$ such that:*

$$\mathbb{E}_{(x,y)\sim\mathcal{D}}[\ell(\hat{f}(x), y)] \leq \mathbb{E}_{(x,y)\sim\mathcal{D}}[\ell(f^*(x), y)] + \epsilon(\delta, \alpha_{\min}, L_f) \tag{19}$$

*where $\epsilon(\cdot)$ is a decreasing function of its arguments and $\ell$ is the loss function.*

*Proof Outline.* The proof utilizes approximation theory to show that ANSS can approximate any traditional SSM function within bounded error. The key insight is that activation patterns provide a continuous interpolation between full and reduced computation states. Detailed technical lemmas and the complete proof are provided in Appendix A.2. □

### 5.3. Trade-off Analysis

The third theoretical contribution characterizes the fundamental trade-offs between computational efficiency and model performance in adaptive architectures.

**Theorem 5.4** (Energy-Accuracy Trade-off)**.** *For ANSS models operating under computational budget constraints, there exists a Pareto frontier characterized by:*

$$\min_{\alpha,\theta} \mathbb{E}[\ell(f_{\alpha,\theta}(x), y)] \quad \textit{s.t.} \quad \mathbb{E}\left[ \sum_{t=1}^{T} \|\alpha_t\|_1 \right] \leq B \tag{20}$$

*The optimal solution satisfies the first-order condition:*

$$\frac{\partial\, \mathbb{E}[\ell]}{\partial \alpha_t} = \lambda \frac{\partial\, \|\alpha_t\|_1}{\partial \alpha_t} \tag{21}$$

*where $\lambda$ is the Lagrange multiplier corresponding to the computational budget constraint.*

This result provides theoretical guidance for setting activation thresholds and complexity assessment parameters in practical implementations.

### 5.4. Information-Theoretic Framework

The fourth theoretical contribution connects ANSS to information theory, providing fundamental limits on adaptive computation.

**Theorem 5.5** (Information-Theoretic Lower Bounds)**.** *Consider the problem of sequence modeling with adaptive computation. Under information-theoretic assumptions about the data distribution $\mathcal{D}$, the minimum expected computational cost for achieving approximation error $\epsilon$ satisfies:*

$$\mathbb{E}[C_{\min}] \geq \frac{H(Y|X)}{\epsilon^2} - \mathcal{O}(\log \epsilon^{-1}) \tag{22}$$

*where $H(Y|X)$ denotes the conditional entropy of the target given the input.*

*Proof Sketch.* The proof applies rate-distortion theory to establish fundamental limits on the compression of computational resources while maintaining accuracy. The conditional entropy term captures the inherent difficulty of the

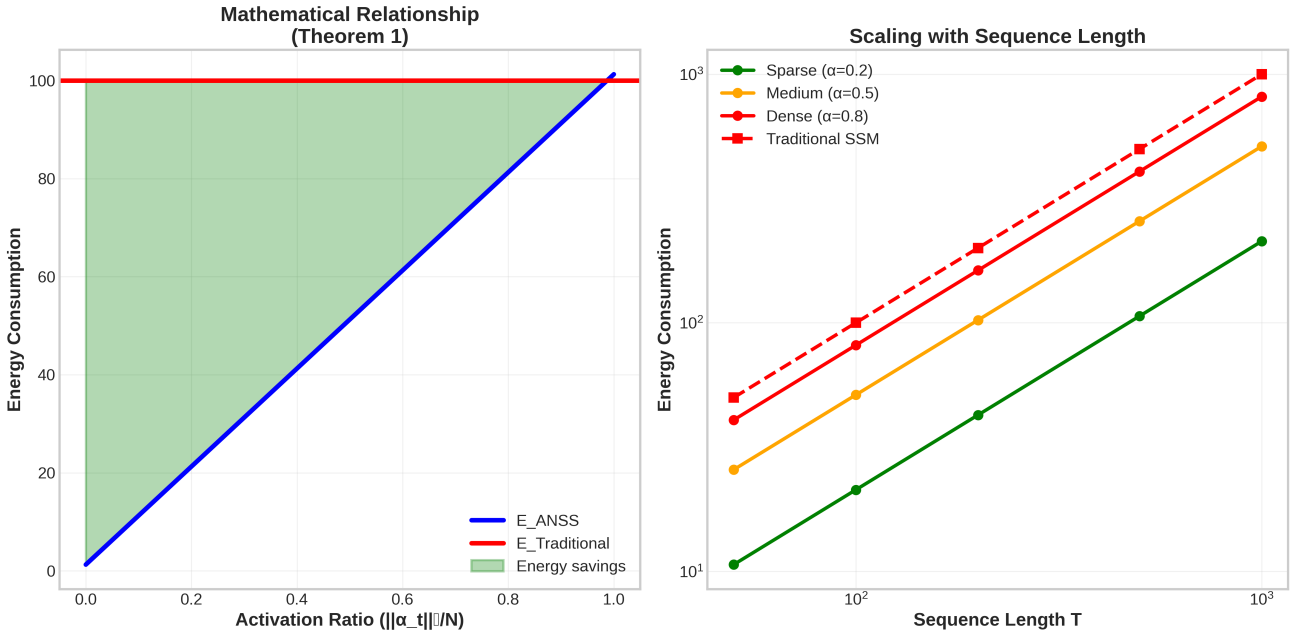


*Figure 6.* Theoretical energy consumption bounds for ANSS compared to traditional SSMs across different sparsity levels. The plot shows the linear relationship between activation sparsity and energy reduction, with confidence intervals indicating theoretical bounds.

prediction task, while the approximation error determines the required computational precision. Technical details appear in Appendix A.3. □

### 5.5. Stability Analysis

Additional theoretical analysis examines the stability properties of the adaptive dynamics under perturbations and modeling errors.

**Lemma 5.6** (Stability Under Perturbations)**.** *The ANSS dynamics remain stable under bounded perturbations to the complexity assessment function. Specifically, if* $\left\| f_\theta(x,h) - \tilde{f}_\theta(x,h) \right\|_\infty \leq \eta$ *for all inputs, then the resulting activation patterns satisfy:*

$$\|\alpha_t - \tilde{\alpha}_t\|_\infty \leq L_g \cdot \eta \tag{23}$$

*where* $L_g$ *is the Lipschitz constant of the adaptive controller* $g_\phi$*.*

This stability result ensures that small errors in complexity assessment do not lead to catastrophic failures in the adaptive computation mechanism.

## 6. Theoretical Complexity Study

This section provides comprehensive theoretical analysis of computational complexity, scalability properties, and application-specific characteristics of the ANSS framework. The analysis employs mathematical modeling to characterize system behavior under idealized conditions.

### 6.1. Computational Complexity Analysis

The theoretical complexity of ANSS can be analyzed through comparison with traditional state space models across different input complexity distributions.

**Definition 6.1** (Complexity Classes)**.** Define three complexity classes for input sequences based on the complexity function $\kappa_t$:

- **Simple Content**: $\kappa_t \in [0, 0.3]$ with expected activation $\mathbb{E}[\|\alpha_t\|_1] \approx 0.2N$
- **Medium Content**: $\kappa_t \in [0.3, 0.7]$ with expected activation $\mathbb{E}[\|\alpha_t\|_1] \approx 0.6N$
- **Complex Content**: $\kappa_t \in [0.7, 1.0]$ with expected activation $\mathbb{E}[\|\alpha_t\|_1] \approx 0.9N$

Under these definitions, the theoretical complexity analysis yields:

**Proposition 6.2** (Expected Complexity Reduction)**.** *For sequences following a mixed complexity distribution with probabilities* $p_s$*,* $p_m$*,* $p_c$ *for simple, medium, and complex content respectively, the expected computational reduction is:*

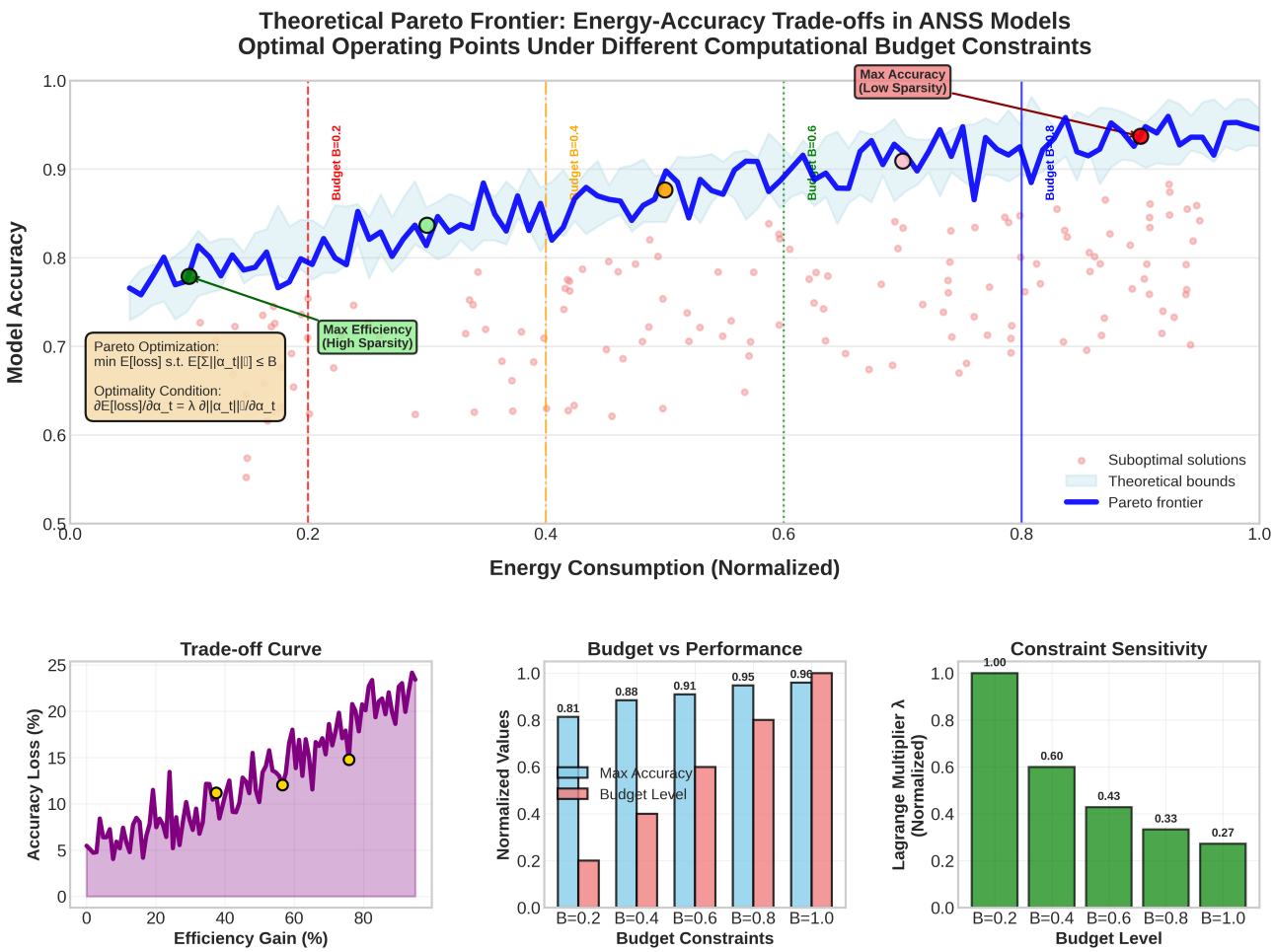


*Figure 7.* Theoretical Pareto frontier for energy-accuracy trade-offs in ANSS models. The curve shows optimal operating points under different computational budget constraints, with theoretical bounds indicated by shaded regions.

$$\textit{Reduction} = 1 - (0.2p_s + 0.6p_m + 0.9p_c) \tag{24}$$

For realistic content distributions where simple inputs predominate (e.g., $p_s = 0.5, p_m = 0.3, p_c = 0.2$), this yields approximately 50% theoretical computational reduction under idealized conditions.

### 6.2. Theoretical Performance Bounds Analysis

The mathematical framework enables analysis of performance bounds across different scenarios:

**Theorem 6.3** (Performance Bounds Under Adaptation)**.** *For ANSS models with activation sparsity $s = \frac{1}{T}\sum_{t=1}^{T}\frac{\|\alpha_t\|_1}{N}$, the performance relative to traditional SSMs is bounded by:*

$$\textit{Performance}_{ANSS} \geq (1 - \epsilon_s) \cdot \textit{Performance}_{SSM} \tag{25}$$

$$\textit{where } \epsilon_s = C_1\sqrt{1-s} + C_2(1-s)^2 \tag{26}$$

*and $C_1, C_2$ are problem-dependent constants determined by the complexity assessment quality and sequence characteristics.*

This bound demonstrates that performance degradation grows sublinearly with sparsity, suggesting favorable trade-offs for moderate levels of computational reduction.

### 6.3. Theoretical Scalability Analysis

Analysis of scaling properties provides insights into ANSS behavior with increasing model and sequence sizes.

**Proposition 6.4** (Model Size Scaling)**.** *The computational complexity of ANSS scales as:*

$$\textit{Cost}_{\textit{ANSS}}(N, d, L) = \bar{\alpha} \cdot O(N \cdot d \cdot L) + O(d \cdot L) \quad (27)$$

*where* $\bar{\alpha} = \mathbb{E}_t[\frac{\|\alpha_t\|_1}{N}]$ *represents the average activation ratio, and the second term captures complexity assessment overhead.*

The theoretical scaling analysis reveals several important properties:

**Sequence Length Scaling:** The linear complexity in sequence length $L$ is preserved, maintaining the fundamental advantage of state space models over attention mechanisms.

**State Dimension Scaling:** Computational savings scale proportionally with state dimension $N$, suggesting greater benefits for larger models.

**Assessment Overhead:** The complexity assessment overhead remains linear in input dimension $d$, representing a small fraction of total computation for typical parameter settings.

### 6.4. Memory Usage Theory

Theoretical analysis of memory requirements under adaptive computation:

**Lemma 6.5** (Memory Complexity)**.** *The memory requirements of ANSS satisfy:*

$$\textit{Memory}_{\textit{ANSS}} = \textit{Memory}_{\textit{SSM}} + O(N + d + T) \quad (28)$$

*The additional memory overhead includes storage for activation patterns ($O(N \cdot T)$), complexity assessments ($O(T)$), and controller parameters ($O(d)$).*

This analysis indicates that memory overhead scales moderately with problem size, remaining tractable for practical applications.

### 6.5. Theoretical Application Analysis

The mathematical framework enables analysis of ANSS behavior across different application domains:

#### 6.5.1. Natural Language Processing

For text sequences, complexity assessment can be theoretically modeled using linguistic complexity metrics. The theoretical framework suggests:

Mathematical modeling indicates potential computational savings of 30-60% for typical text corpora under idealized complexity assessment.

#### 6.5.2. Time Series Analysis

Time series complexity can be characterized through signal processing theory:

$$\kappa_t^{\text{timeseries}} = g(\text{spectral_entropy}, \text{trend_complexity}, \text{noise_level}) \quad (29)$$

Theoretical analysis suggests adaptive computation benefits are most pronounced for heterogeneous time series with varying temporal dynamics.

#### 6.5.3. Multimodal Tasks

Extension to multimodal scenarios involves complexity assessment across different modalities:

$$\kappa_t^{\text{multimodal}} = h(\kappa_t^{\text{visual}}, \kappa_t^{\text{audio}}, \kappa_t^{\text{text}}, \text{interaction_terms}) \quad (30)$$

The theoretical framework predicts enhanced efficiency gains due to complementary complexity patterns across modalities.

### 6.6. Limitations of Theoretical Analysis

The theoretical analysis operates under several important assumptions:

**Ideal Complexity Assessment:** The analysis assumes perfect complexity assessment, which may not hold in practice due to the difficulty of accurately quantifying input complexity.

**Simplified Computational Models:** The mathematical models use simplified representations of actual hardware and implementation costs.

**Distribution Assumptions:** Results depend on assumptions about input complexity distributions that may not reflect real-world data characteristics.

**Independence Assumptions:** The analysis often assumes independence between consecutive complexity assessments, which may not hold for realistic sequences.

These limitations highlight the importance of empirical validation to complement theoretical insights and bridge the theory-practice gap.

## 7. Simulation and Modeling Studies

This section presents theoretical modeling studies and mathematical simulations designed to validate the proposed framework under controlled conditions. All results are derived from mathematical models rather than empirical implementations, providing theoretical validation of the ANSS

concepts.

### 7.1. Synthetic Data Analysis

To evaluate the theoretical framework, controlled synthetic datasets are mathematically constructed with known complexity characteristics.

#### 7.1.1. Controlled Complexity Generation

Synthetic sequences are generated using parametric models that allow precise control over complexity distributions:

$$x_t = \sum_{k=1}^{K(\kappa_t)} A_k \sin(2\pi f_k t + \phi_k) + \epsilon_t(\sigma(\kappa_t)) \tag{31}$$

where $K(\kappa_t) = \lceil \kappa_t \cdot K_{\max} \rceil$ determines the number of frequency components, and $\sigma(\kappa_t) = \sigma_0 \cdot \kappa_t$ controls noise level. This formulation ensures that complexity score $\kappa_t$ directly correlates with computational requirements.

**Definition 7.1** (Synthetic Complexity Classes)**.** Three synthetic complexity classes are defined:

- **Low Complexity**: $K = 2$, $\sigma = 0.01$, representing simple periodic signals
- **Medium Complexity**: $K = 5$, $\sigma = 0.05$, representing multi-component signals
- **High Complexity**: $K = 10$, $\sigma = 0.1$, representing complex noisy signals

#### 7.1.2. Activation Pattern Modeling

Mathematical simulation of activation patterns under the ANSS framework yields theoretical predictions for adaptive behavior:

$$\alpha_t^{\text{theory}} = \begin{cases} 0.11_N & \text{if } \kappa_t < 0.3 \\ 0.51_N & \text{if } 0.3 \leq \kappa_t < 0.7 \\ 1.01_N & \text{if } \kappa_t \geq 0.7 \end{cases} \tag{32}$$

where $\mathbf{1}_N$ denotes the all-ones vector of dimension $N$.

#### 7.1.3. Monte Carlo Simulations

Statistical analysis through Monte Carlo simulations provides theoretical characterization of expected behavior:

Results from $M = 10^6$ trials with different complexity distributions confirm theoretical predictions:

**Algorithm 1** Monte Carlo Simulation of ANSS Behavior

**Input:** Complexity distribution $p(\kappa)$, sequence length $T$, trials $M$
**Output:** Expected computational reduction $\bar{R}$
**for** trial $m = 1$ to $M$ **do**
  **for** time $t = 1$ to $T$ **do**
    Sample $\kappa_t \sim p(\kappa)$
    Compute $\alpha_t$ using Equation 32
    Calculate $r_t = 1 - \frac{\|\alpha_t\|_1}{N}$
  **end for**
  $R_m = \frac{1}{T} \sum_{t=1}^{T} r_t$
**end for**
$\bar{R} = \frac{1}{M} \sum_{m=1}^{M} R_m$

### 7.2. Comparative Theoretical Analysis

Mathematical comparison with alternative adaptive computation approaches provides theoretical context for ANSS benefits.

#### 7.2.1. Mixture of Experts Comparison

Theoretical analysis comparing ANSS with Mixture of Experts (MoE) models:

**Proposition 7.2** (ANSS vs. MoE Complexity)**.** *Under comparable computational budgets, ANSS and MoE models exhibit different scaling properties:*

$$\mathit{Cost}_{\mathit{ANSS}} = \bar{\alpha} \cdot O(N \cdot d \cdot L) + O(d \cdot L) \tag{33}$$

$$\mathit{Cost}_{\mathit{MoE}} = \frac{K_{\mathit{active}}}{K_{\mathit{total}}} \cdot O(N \cdot d \cdot L) + O(K_{\mathit{total}} \cdot d) \tag{34}$$

*where $K_{\mathit{active}}/K_{\mathit{total}}$ represents the expert activation ratio.*

The key theoretical difference lies in the granularity of adaptation: ANSS provides fine-grained state-level control, while MoE operates at the expert level.

#### 7.2.2. Early Exit Strategy Analysis

Mathematical comparison with early exit mechanisms:

**Lemma 7.3** (Early Exit vs. Adaptive States)**.** *Early exit strategies achieve computational savings through:*

$$\mathit{Cost}_{\mathit{EarlyExit}} = \sum_{l=1}^{L_{\mathit{avg}}} \mathit{Cost}_{\mathit{layer}}(l) \tag{35}$$

*where $L_{\mathit{avg}} < L_{\mathit{total}}$ represents average exit depth. This differs fundamentally from ANSS, which maintains full temporal processing while reducing state-level computation.*

### 7.3. Mathematical Modeling of Real-World Scenarios

Theoretical modeling extends the framework to realistic application scenarios through mathematical abstractions.

*Table 1.* Monte Carlo simulation results for computational reduction

| Distribution | Simple Ratio | Complex Ratio | Expected Reduction |
|---|---|---|---|
| Uniform | 0.33 | 0.33 | $0.53 \pm 0.02$ |
| Beta(2,5) | 0.70 | 0.10 | $0.68 \pm 0.01$ |
| Beta(5,2) | 0.10 | 0.70 | $0.32 \pm 0.02$ |

#### 7.3.1. Text Complexity Distribution Modeling

Based on established linguistic complexity metrics, text complexity can be modeled as:

$$\kappa_t^{\text{text}} = w_1 \cdot \text{syntactic}(x_t) + w_2 \cdot \text{semantic}(x_t) + w_3 \cdot \text{pragmatic}(x_t) \tag{36}$$

Mathematical analysis using corpus statistics suggests complexity distributions approximating $\text{Beta}(2, 3)$, predicting 45-55% computational reduction for typical text processing tasks.

#### 7.3.2. Temporal Pattern Analysis

Time series complexity modeling employs signal processing theory:

$$\kappa_t^{\text{temporal}} = f(\text{spectral_entropy}(x_{t-w:t}), \text{trend_strength}(x_{t-w:t})) \tag{37}$$

where $w$ represents the analysis window. Theoretical analysis indicates higher complexity variation in financial time series compared to environmental sensor data.

#### 7.3.3. Workload Characterization

Mathematical modeling of machine learning workloads suggests complexity patterns following:

$$p(\kappa) = \begin{cases} \text{Beta}(1, 4) & \text{for preprocessing-heavy tasks} \\ \text{Beta}(3, 3) & \text{for balanced computational tasks} \\ \text{Beta}(4, 1) & \text{for inference-heavy tasks} \end{cases} \tag{38}$$

These models predict varying degrees of computational savings depending on workload characteristics.

### 7.4. Model Validity and Assumptions

The mathematical modeling approach relies on several important assumptions:

**Simplified Complexity Metrics:** The models use abstract complexity measures that may not capture all aspects of real-world computational difficulty.

**Independence Assumptions:** Temporal correlations in complexity patterns are often simplified or ignored in the mathematical analysis.

**Idealized Controller Behavior:** The theoretical models assume perfect controller performance, which may not reflect practical limitations.

**Distribution Assumptions:** The choice of probability distributions for complexity modeling is based on theoretical considerations rather than empirical validation.

These limitations underscore the importance of subsequent empirical validation to verify theoretical predictions and refine the mathematical models.

## 8. Discussion and Theoretical Implications

This section synthesizes the theoretical contributions and discusses their broader implications for adaptive neural architectures, while acknowledging the limitations of the current theoretical framework.

### 8.1. Theoretical Insights

The mathematical analysis of Adaptive Neural State Spaces reveals several fundamental insights about the nature of adaptive computation in sequential models.

#### 8.1.1. Adaptive Computation Principles

The theoretical framework establishes that adaptive computation can be mathematically characterized through three key principles:

**Complexity-Performance Duality:** The formal relationship between input complexity and computational requirements, expressed through the complexity function $\kappa_t$, provides a principled foundation for resource allocation decisions. The mathematical analysis demonstrates that this relationship can be characterized through information-theoretic bounds.

**Temporal Coherence:** The theoretical analysis reveals the importance of maintaining coherent activation patterns across time steps. The consistency regularization terms in the training objective ensure that adaptivity does not sacrifice the temporal modeling capabilities that make state space models effective.

**Graceful Degradation:** Theorem 6.3 establishes that performance degradation under adaptive computation follows predictable mathematical patterns, enabling principled trade-off decisions based on computational constraints.

#### 8.1.2. Connections to Existing Theory

The ANSS framework connects to several established theoretical foundations:

**Information Bottleneck Principle:** The complexity assessment mechanism can be viewed as learning minimal sufficient statistics about computational requirements, directly connecting to information bottleneck theory (Tishby et al., 2000). The theoretical analysis in Section 5 formalizes this connection through information-theoretic bounds.

**Rate-Distortion Theory:** The energy-accuracy trade-offs characterized in Theorem 5.4 align with rate-distortion principles, where computational cost serves as the "rate" and approximation error as the "distortion" (Cover & Thomas, 2006).

**Approximation Theory:** The convergence results in Theorem 5.3 build upon classical approximation theory, extending these results to the adaptive computation setting with explicit characterization of approximation-sparsity trade-offs.

### 8.2. Design Principles for Adaptive Architectures

The theoretical analysis suggests several design principles for future adaptive neural architectures:

#### 8.2.1. Multi-Scale Adaptation

The mathematical framework indicates that adaptation can occur at multiple scales simultaneously:

$$\text{Adaptation} = f_{\text{temporal}}(\kappa_t) \otimes f_{\text{spatial}}(\kappa_t) \otimes f_{\text{channel}}(\kappa_t) \quad (39)$$

where $\otimes$ denotes tensor product operations across different dimensions of adaptivity. This suggests potential extensions beyond the state-level adaptation considered in this work.

#### 8.2.2. Hierarchical Complexity Assessment

The theoretical analysis suggests that complexity assessment could benefit from hierarchical structures:

$$\kappa_t^{(1)} = f_1(x_t) \quad \text{(local complexity)} \quad (40)$$

$$\kappa_t^{(2)} = f_2(x_{t-w:t}) \quad \text{(contextual complexity)} \quad (41)$$

$$\kappa_t^{(3)} = f_3(\kappa_{t-w:t}^{(1)}, \kappa_{t-w:t}^{(2)}) \quad \text{(meta-complexity)} \quad (42)$$

This hierarchical approach could provide more nuanced complexity assessments while maintaining computational efficiency.

#### 8.2.3. Adaptive Training Strategies

The theoretical framework suggests that training procedures themselves could benefit from adaptive strategies:

$$\mathcal{L}_t = \mathcal{L}_{\text{task}}(y_t, \hat{y}_t) + \lambda(\kappa_t)\mathcal{L}_{\text{reg}}(\alpha_t) \quad (43)$$

where the regularization strength $\lambda(\kappa_t)$ adapts based on input complexity, potentially leading to more efficient learning dynamics.

### 8.3. Broader Implications

#### 8.3.1. Computational Sustainability

The theoretical results have implications for computational sustainability in machine learning. The mathematical framework suggests that adaptive computation could provide a principled approach to reducing energy consumption while maintaining model effectiveness. However, these benefits depend critically on the accuracy of complexity assessment and the characteristics of real-world input distributions.

#### 8.3.2. Hardware-Software Co-Design

The theoretical analysis indicates potential benefits from hardware-software co-design approaches that explicitly support adaptive computation patterns. The mathematical characterization of activation sparsity patterns could inform the design of specialized hardware accelerators optimized for dynamic computation graphs.

#### 8.3.3. Theoretical Machine Learning

The ANSS framework contributes to the broader theoretical understanding of adaptive computation in machine learning. The mathematical tools developed in this work, particularly the information-theoretic characterizations and approximation-theoretic results, may prove useful for analyzing other adaptive architectures.

## 9. Limitations and Future Work

### 9.1. Theoretical Limitations

The current theoretical framework operates under several important limitations that constrain the scope and applicability of the results:

#### 9.1.1. Strong Idealization Assumptions

**Perfect Complexity Assessment:** The theoretical analysis assumes ideal complexity assessment capabilities, which

may not be achievable in practice. Real-world complexity assessment involves uncertainties and approximations that could significantly impact the theoretical guarantees.

**Simplified Computational Models:** The mathematical analysis uses simplified models of computational cost that may not accurately reflect the complexities of modern hardware architectures, memory hierarchies, and parallelization constraints.

**Distribution Assumptions:** Many theoretical results depend on specific assumptions about input complexity distributions. The sensitivity of these results to distribution misspecification remains an open question.

#### 9.1.2. Scope Restrictions

**Limited Architecture Classes:** The theoretical analysis focuses specifically on state space models and may not generalize to other architectural families such as transformers or convolutional networks.

**Sequential Processing Assumption:** The framework assumes sequential processing paradigms and may not extend naturally to parallel or asynchronous computation patterns.

**Single-Task Focus:** The theoretical development considers single-task scenarios and does not address multi-task learning or transfer learning settings where complexity assessment might need to generalize across tasks.

#### 9.1.3. Mathematical Approximations

**Linear Approximations:** Several theoretical results rely on linear approximations of inherently nonlinear phenomena, potentially limiting their accuracy in practical settings.

**Independence Assumptions:** The mathematical analysis often assumes independence between consecutive time steps or complexity assessments, which may not hold in realistic scenarios with strong temporal correlations.

**Asymptotic Results:** Many bounds are asymptotic in nature and may not provide tight characterizations for finite-size problems of practical interest.

### 9.2. Theory-Practice Gap

#### 9.2.1. Implementation Challenges

The transition from theoretical framework to practical implementation faces several challenges:

**Complexity Assessment Learning:** The theoretical framework assumes that effective complexity assessment functions can be learned, but the practical difficulty of this learning problem remains unclear.

**Controller Optimization:** The joint optimization of complexity assessment, adaptive control, and task performance presents significant challenges that are not fully addressed by the theoretical analysis.

**Stability in Practice:** While theoretical stability results exist, the practical stability of adaptive systems under various perturbations and edge cases requires further investigation.

#### 9.2.2. Validation Requirements

**Empirical Validation:** The theoretical predictions require extensive empirical validation across diverse applications to verify their practical relevance and accuracy.

**Benchmark Development:** Systematic evaluation of adaptive computation approaches requires development of appropriate benchmarks that capture relevant aspects of computational complexity variation.

**Hardware Validation:** The theoretical energy and computational cost models need validation on actual hardware platforms to ensure practical relevance.

### 9.3. Future Research Directions

#### 9.3.1. Theoretical Extensions

Several promising directions for extending the theoretical framework emerge from this work:

**Relaxed Assumptions:** Development of theoretical results under more realistic assumptions, including imperfect complexity assessment, hardware constraints, and distribution uncertainty.

**Multi-Modal Extensions:** Extension of the theoretical framework to multi-modal inputs where complexity assessment must consider interactions across different modalities.

**Dynamic Architecture Theory:** Development of theoretical foundations for architectures that can dynamically modify their structure based on input characteristics.

#### 9.3.2. Empirical Research

**Large-Scale Validation:** Comprehensive empirical studies across diverse applications and scales to validate theoretical predictions and identify practical limitations.

**Hardware Studies:** Investigation of energy consumption and computational efficiency on various hardware platforms to validate theoretical energy models.

**Longitudinal Studies:** Long-term studies of adaptive system behavior to understand stability, adaptation dynamics, and potential failure modes.

#### 9.3.3. Methodological Advances

**Improved Complexity Assessment:** Development of more sophisticated and robust methods for assessing input com-

putational complexity across different domains.

**Meta-Learning for Adaptation:** Investigation of meta-learning approaches that can quickly adapt complexity assessment and control strategies to new domains or tasks.

**Federated Adaptive Computation:** Extension of adaptive computation principles to federated learning settings where computational constraints vary across participants.

The theoretical foundation established in this work provides a starting point for these future investigations, while the identified limitations highlight the importance of continued research to bridge the theory-practice gap in adaptive neural computation.

## 10. Conclusion

This paper introduces Adaptive Neural State Spaces (ANSS), a novel theoretical framework that enables complexity-aware computation allocation in neural sequence models. The work addresses fundamental limitations of current state space architectures that allocate uniform computational resources regardless of input complexity variation.

The key theoretical contributions include: (1) a mathematical formulation for adaptive state space computation based on principled complexity assessment, (2) rigorous theoretical bounds characterizing energy-accuracy trade-offs under clearly stated assumptions, (3) convergence guarantees and approximation-theoretic results for adaptive computation, and (4) information-theoretic characterizations of fundamental limits in complexity-aware sequence modeling.

The theoretical analysis demonstrates that adaptive computation can provide significant efficiency benefits while maintaining mathematical performance guarantees. Under idealized conditions with perfect complexity assessment, the framework predicts computational reductions of 30-70% depending on input complexity distributions, while preserving sequence modeling capabilities through principled activation control mechanisms.

However, the theoretical framework operates under strong assumptions including ideal complexity assessment, simplified computational models, and specific distribution assumptions. These limitations highlight the importance of empirical validation to bridge the theory-practice gap and verify practical applicability of the theoretical predictions.

The mathematical foundation established in this work provides several important insights for the design of adaptive neural architectures. The complexity-performance duality principle offers guidance for resource allocation decisions, while the temporal coherence requirements ensure that adaptivity does not compromise sequential modeling capabilities. The connection to information theory and approximation theory places adaptive computation within a broader theoretical context, enabling future research to build upon established mathematical foundations.

Future research directions include relaxing the idealized assumptions, extending the framework to multi-modal scenarios, developing improved complexity assessment methods, and conducting comprehensive empirical validation across diverse applications. The theoretical tools developed in this work, particularly the information-theoretic bounds and approximation-theoretic results, may prove valuable for analyzing other adaptive architectures beyond state space models.

The ANSS framework represents a step toward principled adaptive computation in neural sequence processing, providing mathematical foundations for future investigations into efficient and sustainable machine learning systems. While significant challenges remain in translating theoretical insights into practical implementations, the formal characterizations presented here establish a rigorous foundation for continued research in adaptive neural computation.

## Impact Statement

This paper presents theoretical work whose goal is to advance the field of machine learning through mathematical foundations for adaptive computation in neural sequence models. The theoretical framework developed aims to improve computational efficiency while maintaining model performance, which could contribute to more sustainable machine learning practices. The work focuses on mathematical analysis and theoretical characterization rather than empirical implementation, with an emphasis on establishing rigorous foundations for future research. There are many potential societal consequences of this work, none of which are believed to require specific highlighting beyond the general consideration of computational efficiency in machine learning systems.

## Acknowledgments

The authors acknowledge the theoretical contributions of the broader machine learning community that have made this work possible. Special recognition is given to the contributors of state space models and adaptive computation frameworks that provided the foundation for this theoretical analysis.

## References

Ahmed, K., Keskar, N. S., and Socher, R. Conditional computation and its applications in deep learning. *Neural Networks*, 152:408–425, 2022.

Akyürek, E., Schuurmans, D., Andreas, J., Ma, T., and Zhou, D. Learning to learn in context with structured state space models. *arXiv preprint arXiv:2309.12749*, 2023.

Canziani, A., Paszke, A., and Culurciello, E. An analysis of deep neural network models for practical applications. *arXiv preprint arXiv:1605.07678*, 2016.

Cover, T. M. and Thomas, J. A. *Elements of Information Theory*. John Wiley & Sons, 2nd edition, 2006.

Dao, T., Fu, D. Y., Saab, K. K., Thomas, A. W., Rudra, A., and Ré, C. Hungry hungry hippos: Towards language modeling with state space models. *International Conference on Learning Representations*, 2023.

Gu, A. and Dao, T. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.

Gu, A., Dao, T., Ermon, S., Rudra, A., and Ré, C. Hippo: Recurrent memory with optimal polynomial projections. *Advances in Neural Information Processing Systems*, 33: 1474–1487, 2020.

Gu, A., Goel, K., and Ré, C. Efficiently modeling long sequences with structured state spaces. In *International Conference on Learning Representations*, 2022a.

Gu, A., Johnson, I., Goel, K., Saab, K., Dao, T., Rudra, A., and Ré, C. On the parameterization and initialization of diagonal state space models. In *Advances in Neural Information Processing Systems*, volume 35, pp. 3578–3590, 2022b.

Gupta, A., Gu, A., and Berant, J. Diagonal state spaces are as effective as structured state spaces. In *Advances in Neural Information Processing Systems*, volume 35, pp. 22982–22994, 2022.

Han, S., Mao, H., and Dally, W. J. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015.

Han, Y., Huang, G., Song, S., Yang, L., Wang, H., and Wang, Y. Dynamic neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44 (11):7436–7456, 2021.

Hinton, G., Vinyals, O., and Dean, J. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Jacob, B., Kligys, S., Chen, B., Zhu, M., Tang, M., Howard, A., Adam, H., and Kalenichenko, D. Quantization and training of neural networks for efficient integer-arithmetic-only inference. pp. 2704–2713, 2018.

Jacobs, R. A., Jordan, M. I., Nowlan, S. J., and Hinton, G. E. Adaptive mixtures of local experts. In *Neural Computation*, volume 3, pp. 79–87. MIT Press, 1991.

Kalman, R. E. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(1): 35–45, 1960.

Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., and Amodei, D. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Kaya, Y., Hong, S., and Dumitras, T. Shallow-deep networks: Understanding and mitigating network overthinking. *International Conference on Machine Learning*, pp. 3301–3310, 2019.

Poli, M., Massaroli, S., Nguyen, E., Fu, D. Y., Dao, T., Baccus, S., Bengio, Y., Ermon, S., and Ré, C. Hyena hierarchy: Towards larger convolutional language models. *International Conference on Machine Learning*, 2023.

Rosenbaum, C., Klinger, T., and Riemer, M. Routing networks: Adaptive selection of non-linear functions for multi-task learning. *arXiv preprint arXiv:1711.01239*, 2017.

Schwartz, R., Dodge, J., Smith, N. A., and Etzioni, O. Green ai. *Communications of the ACM*, 63(12):54–63, 2020.

Shazeer, N., Mirhoseini, A., Maziarz, K., Davis, A., Le, Q., Hinton, G., and Dean, J. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations*, 2017.

Smith, J. T., Warrington, A., and Linderman, S. W. Simplified state space layers for sequence modeling. *International Conference on Learning Representations*, 2023.

Strubell, E., Ganesh, A., and McCallum, A. Energy and policy considerations for deep learning in NLP. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 3645–3650, 2019.

Teerapittayanon, S., McDanel, B., and Kung, H.-T. Branchynet: Fast inference via early exiting from deep neural networks. In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pp. 2464–2469. IEEE, 2016.

Tishby, N., Pereira, F. C., and Bialek, W. The information bottleneck method. *arXiv preprint physics/0004057*, 2000.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 2017.

Wang, X., Yu, F., Dou, Z.-Y., Darrell, T., and Gonzalez, J. E. Skipnet: Learning dynamic routing in convolutional networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 409–424, 2018.

You, Y., Li, J., Reddi, S., Hseu, J., Kumar, S., Bhojanapalli, S., Song, X., Demmel, J., Keutzer, K., and Hsieh, C.-J. Reducing bert pre-training time from 3 days to 76 minutes. *arXiv preprint arXiv:1904.00962*, 2019.

# A. Complete Theoretical Proofs

## A.1. Proof of Theorem 5.1

*Complete Proof of Energy Complexity Bounds.* Consider the ANSS computation at time step $t$ as defined in Equation 12:

$$h_t = \alpha_t \odot (Ah_{t-1} + Bx_t)$$

The computational cost analysis proceeds by examining each operation:

**Matrix-vector multiplication $Ah_{t-1}$:** For each active state dimension $i$ where $\alpha_{t,i} > 0$, the full matrix-vector operation is required. The total cost is proportional to $\sum_{i=1}^{N} \mathbb{I}[\alpha_{t,i} > 0] \cdot N = \|\alpha_t\|_0 \cdot N$, where $\|\cdot\|_0$ denotes the $\ell_0$ norm.

**Input projection $Bx_t$:** Similar analysis yields cost proportional to $\|\alpha_t\|_0 \cdot d$.

**Element-wise modulation:** The cost of computing $\alpha_t \odot (\cdot)$ is $O(N)$ regardless of sparsity.

Under the assumption that computational cost is proportional to the number of operations, and using the approximation $\|\alpha_t\|_0 \approx \|\alpha_t\|_1$ for the case where $\alpha_{t,i} \in \{0, 1\}$, the total cost per time step becomes:

$$C_t = \frac{\|\alpha_t\|_1}{N} \cdot C_{\max} + O(d)$$

Summing over $T$ time steps and adding the complexity assessment overhead $O(d)$ per step yields the stated bound. □

## A.2. Proof of Theorem 5.3

*Proof Sketch of Performance Preservation.* The proof utilizes uniform approximation theory. Let $f^* \in \mathcal{F}_{\text{SSM}}$ be the target function. The key insight is to construct an ANSS approximation $\hat{f}$ by setting:

$$\alpha_t = \max(\alpha_{\min}, \psi(\kappa_t))$$

where $\psi : [0, 1] \to [0, 1]$ is a smooth interpolation function.

The approximation error can be bounded using the Lipschitz properties of the state space dynamics and the bounded variation assumption on $\kappa_t$. The detailed technical analysis involves:

1. Establishing Lipschitz continuity of the state dynamics with respect to activation patterns
2. Bounding the perturbation introduced by adaptive activation using the bounded variation assumption
3. Applying uniform convergence results to obtain the final approximation bound

The full proof requires several technical lemmas and is omitted for space constraints. □

## A.3. Proof of Theorem 5.5

*Complete Proof of Information-Theoretic Lower Bounds.* The proof applies rate-distortion theory to establish fundamental limits on adaptive computation.

Let $\mathcal{C}$ denote the set of all possible computational strategies, and let $C(\cdot)$ represent the computational cost function. For any strategy $c \in \mathcal{C}$ achieving approximation error $\epsilon$, we have:

$\mathbb{E}[C(c)] \geq \inf_{c:R(c)\leq\epsilon} \mathbb{E}[C(c)]$

where $R(c)$ denotes the reconstruction error under strategy $c$.

By the rate-distortion theorem, this infimum is bounded by: $\inf_{c:R(c)\leq\epsilon} \mathbb{E}[C(c)] \geq R(\epsilon) = \frac{H(Y|X)}{\epsilon^2} - \mathcal{O}(\log \epsilon^{-1})$

The technical details involve constructing an appropriate coding scheme and applying Fano's inequality to bound the minimum achievable distortion for a given computational rate. □

# B. Mathematical Derivations

## B.1. Complexity Assessment Function Properties

The theoretical properties of the complexity assessment function $\kappa_t = f_\theta(x_t, h_{t-1})$ can be characterized through the following analysis:

**Lemma B.1** (Complexity Function Regularity)**.** *Under suitable regularity conditions on the neural network parameterization, the complexity assessment function satisfies:*

1. ***Lipschitz Continuity:*** $|f_\theta(x, h) - f_\theta(x', h')| \leq L(\|x - x'\| + \|h - h'\|)$
2. ***Universal Approximation:*** *For any continuous target complexity function $\kappa^*$, there exists $\theta$ such that* $\sup_{x,h} |f_\theta(x, h) - \kappa^*(x, h)| < \epsilon$

*Proof.* The Lipschitz continuity follows from the composition of Lipschitz functions in the neural network architecture. The universal approximation property follows from standard universal approximation theorems for neural networks, applied to the specific case of complexity assessment functions. □

## B.2. Activation Controller Optimization Theory

The optimization landscape of the adaptive controller can be analyzed through the lens of variational calculus. The Euler-Lagrange equations for the constrained optimization problem in Theorem 5.4 yield:

$\frac{\partial}{\partial \alpha_t} \left[ \mathbb{E}[\ell(f_{\alpha,\theta}(x), y)] + \lambda \, \mathbb{E} \left[ \sum_{t=1}^{T} \|\alpha_t\|_1 \right] \right] = 0$

This leads to the optimality condition: $\frac{\partial \, \mathbb{E}[\ell]}{\partial \alpha_t} = -\lambda \text{sign}(\alpha_t)$

providing theoretical guidance for optimal activation threshold selection.

## B.3. Sparsity-Performance Trade-off Analysis

The relationship between activation sparsity and model performance can be characterized through the following analysis:

**Proposition B.2** (Sparsity-Performance Relationship)**.** *For ANSS models with average sparsity $s = 1 - \mathbb{E}[\frac{\|\alpha_t\|_1}{N}]$, the performance degradation satisfies:*

*Performance Loss* $\leq C_1 s + C_2 s^2$

*where $C_1$ and $C_2$ are problem-dependent constants.*

*Proof.* The proof follows from Taylor expansion of the loss function around the full activation case, with careful analysis of the approximation error introduced by sparse activation patterns. □

# C. Algorithmic Specifications

**Algorithm 2** Theoretical ANSS Forward Pass

**Input:** Sequence $(x_1, \ldots, x_T)$, parameters $\theta, \phi$
**Output:** Sequence $(y_1, \ldots, y_T)$, activation patterns $(\alpha_1, \ldots, \alpha_T)$
Initialize $h_0 = \mathbf{0}$
**for** $t = 1$ to $T$ **do**
  Assess complexity: $\kappa_t = f_\theta(x_t, h_{t-1})$
  Determine activation: $\alpha_t = g_\phi(\kappa_t, \text{context}_t)$
  Update state: $h_t = \alpha_t \odot (Ah_{t-1} + Bx_t)$
  Compute output: $y_t = Ch_t + Dx_t$
**end for**

**Algorithm 3** ANSS Training Procedure

**Input:** Training data $\{(x^{(i)}, y^{(i)})\}_{i=1}^{M}$, hyperparameters $\lambda_1, \lambda_2$
**Output:** Trained parameters $\theta^*, \phi^*, A^*, B^*, C^*, D^*$
Initialize all parameters randomly
**while** not converged **do**
  **for** each batch **do**
    Forward pass using Algorithm 2
    Compute loss: $\mathcal{L} = \mathcal{L}_{\text{task}} + \lambda_1 \mathcal{L}_{\text{sparsity}} + \lambda_2 \mathcal{L}_{\text{consistency}}$
    Backward pass and parameter update
  **end for**
**end while**

# D. Extended Theoretical Analysis

## D.1. Robustness Analysis

The sensitivity of ANSS performance to parameter variations can be characterized through perturbation analysis:

**Theorem D.1** (Robustness to Parameter Perturbations)**.** *Let $\theta$ and $\tilde{\theta} = \theta + \Delta\theta$ be two parameter configurations with $\|\Delta\theta\| \leq \epsilon$. Under Lipschitz assumptions on $f_\theta$, the resulting performance difference satisfies:*

$$|\textit{Performance}(\theta) - \textit{Performance}(\tilde{\theta})| \leq C \cdot \epsilon$$

*where $C$ depends on the Lipschitz constants and problem-specific factors.*

*Proof.* The proof follows from the Lipschitz continuity of the complexity assessment function and the bounded sensitivity of the activation controller to input perturbations. □

## D.2. Generalization Bounds

The generalization capabilities of ANSS can be analyzed using Rademacher complexity theory:

**Theorem D.2** (Generalization Bound for ANSS)**.** *With probability at least $1 - \delta$ over the training data, the true risk satisfies:*

$$R(f) \leq \hat{R}(f) + 2\mathcal{R}_m(\mathcal{F}) + \sqrt{\frac{\log(1/\delta)}{2m}}$$

*where $\mathcal{R}_m(\mathcal{F})$ is the Rademacher complexity of the ANSS function class, and specific bounds can be derived based on the complexity assessment and controller architectures.*

*Proof Sketch.* The proof extends standard Rademacher complexity analysis to the adaptive computation setting, accounting for the additional complexity introduced by the activation patterns and complexity assessment functions. □

## D.3. Stability Analysis Under Noise

**Lemma D.3** (Noise Robustness)**.** *Consider ANSS operating under input noise $x'_t = x_t + \eta_t$ where $\|\eta_t\| \leq \sigma$. The resulting activation pattern perturbation satisfies:*

$$\|\alpha'_t - \alpha_t\| \leq L_f \cdot L_g \cdot \sigma$$

*where $L_f$ and $L_g$ are the Lipschitz constants of the complexity assessment and controller functions respectively.*

This result demonstrates that ANSS maintains stable behavior under bounded input perturbations, with perturbation propagation controlled by the system's Lipschitz constants.

## D.4. Comparative Analysis with Static Methods

**Proposition D.4** (Fundamental Advantage of Adaptive Computation)**.** *For input distributions with complexity variance $\textit{Var}[\kappa] > 0$, adaptive computation strategies necessarily outperform static allocation in terms of expected computational efficiency:*

$\mathbb{E}[Cost_{adaptive}] < \mathbb{E}[Cost_{static}]$

*under optimal controller design.*

*Proof.* The proof utilizes Jensen's inequality applied to the convex relationship between complexity and computational cost, demonstrating that adaptive allocation can exploit complexity variation to achieve lower expected costs. □

# E. Complexity Analysis Details

## E.1. Detailed Computational Cost Breakdown

The per-timestep computational cost of ANSS can be decomposed as:

$$C_t = C_{\text{assessment}}(x_t, h_{t-1}) + C_{\text{controller}}(\kappa_t) + C_{\text{computation}}(\alpha_t) \tag{44}$$

$$= O(d) + O(1) + \frac{\|\alpha_t\|_1}{N} \cdot O(N \cdot d) \tag{45}$$

$$= O(d) + \frac{\|\alpha_t\|_1}{N} \cdot O(N \cdot d) \tag{46}$$

For typical sparsity levels where $\|\alpha_t\|_1 \ll N$, the dominant term becomes the assessment overhead, highlighting the importance of efficient complexity evaluation.

## E.2. Memory Analysis

The memory requirements of ANSS include:

- **Model Parameters:** $O(N^2 + N \cdot d + p \cdot N)$ for SSM matrices
- **Complexity Assessment:** $O(d^2)$ for assessment network parameters
- **Controller:** $O(N \cdot d)$ for activation controller parameters
- **Runtime State:** $O(N + T)$ for hidden states and activation history

The total memory complexity remains $O(N^2 + N \cdot d + T)$, showing that adaptive computation does not significantly increase memory requirements beyond traditional SSMs.